Exploring Side-Channel Vulnerabilities in Homomorphic Encryption: Threats and Countermeasures

Farheen
Student
Masters in Cyber Forensics and Information Security,
Centre for Cyber Forensics and Information Security,
University of Madras, Chennai - 600005
farheen.unom@gmail.com

Premanand Narasimhan
Director, Techiespeaks OPC Pvt Ltd
Independent Researcher/Consultant
Vice President Cyber Society of India
premvn@gmail.com

Dr.N.Kala
Assistant Professor
Former Director i/c
Centre for Cyber Forensics and Information Security
University of Madras,
Chennai – 600005
kalabaskar@gmail.com

Abstract

Homomorphic encryption (HE) allows for numeric data to be manipulated without decrypting the data and the need for encryption keys, preserving privacy in sensitive applications. However, it is possible to break homomorphic encryption using side-channel attacks, which exploit physical or behavioral leakages caused during data transmission. This paper explores these vulnerabilities in homomorphic encryption highlighting critical vulnerabilities in popular HE libraries. Furthermore, this paper will also analyze the effectiveness of existing countermeasures, and propose strategies to enhance HE resilience against such attacks. The findings aim to bridge the gap between theoretical HE security and its practical implementation, fostering robust cryptographic systems in real-world applications.

Keywords: Homomorphic Encryption (HE), Side-Channel Attacks (SCA), Cryptographic vulnerabilities, Data Masking, Timing Attacks, Threat Analysis

Introduction

Homomorphic encryption (HE) is a transformative cryptographic technique that permits computations on encrypted numeric data without decryption, thereby preserving data confidentiality throughout processing. This capability is particularly advantageous in scenarios where sensitive information must be processed in untrusted environments, such as cloud computing and outsourced data services.

Despite HE's robust theoretical security foundations, practical implementations are susceptible to side-channel attacks, exploits that leverage unintended information leakage from cryptographic systems' physical implementations. These attacks can extract sensitive information by analyzing various physical parameters, including power

consumption, electromagnetic emissions, and execution timing.

Recent studies have highlighted the vulnerability of HE schemes to such attacks. Aydin et al. demonstrated that the Microsoft SEAL HE library is prone to side-channel attacks and is capable of extracting encryption keys during the key generation phase by exploiting leakage in the number theoretic transform algorithm. Similarly, another study revealed that power-based side-channel attacks could extract entire plaintext messages from SEAL's encryption phase with a single power measurement. These findings underscore the critical need to analyze and mitigate side-channel vulnerabilities in HE systems to ensure their security in real-world applications.

This paper aims to:

- Explore the susceptibility of homomorphic encryption (HE) schemes to side-channel attacks, providing a comprehensive analysis of potential threats.
- Evaluate existing countermeasures to determine their effectiveness in mitigating these vulnerabilities.
- Propose strategies to enhance the resilience of HE implementations against side-channel attacks, bridging the gap between theoretical security and practical deployment.

By addressing these objectives, we contribute to the development of more secure cryptographic systems suitable for deployment in diverse and potentially adversarial environments.

1. Literature Review

A series of works by Aydin et al. (2020, 2021) have demonstrated several vulnerabilities in the Microsoft SEAL HE library, one of the most widely used libraries for homomorphic encryption. In their paper *Leaking Secrets in Homomorphic Encryption with Side-Channel Attacks*, the authors highlighted how side-channel leakages could allow attackers to extract plaintext messages from encrypted data by analyzing device-side encryption operations. The study emphasized that the encryption operations in SEAL could inadvertently reveal valuable information, making the system susceptible to various forms of SCAs (Aydin et al., 2020).

A subsequent study by Aydin et al., *Exposing Side-Channel Leakage of SEAL Homomorphic Encryption Library*, further investigated the vulnerabilities of SEAL, revealing a new side-channel leakage in which attackers could exploit ternary values leaked during computations. This vulnerability posed a direct threat to the confidentiality of encrypted data, urging the need for countermeasures to mitigate such risks (Aydin et al., 2021).

In another paper, *Single-Trace Side-Channel Leakage of the SEAL Homomorphic Encryption Library*, the authors demonstrated the effectiveness of a side-channel attack on SEAL, which could extract plaintext messages using just a single trace of side-channel information. This work introduced a new level of complexity in side-channel attacks, suggesting that even minimal data leakage could lead to the recovery of sensitive information, further emphasising the vulnerability of HE libraries to SCAs (Aydin et al., 2020).

Finally, the study *Characterizing Memory Side Channels in HE Applications* by Pal et al. (2022) [2] explored the impact of memory access patterns in Fully Homomorphic

Encryption (FHE) applications. The authors found that variations in memory access could unintentionally expose sensitive information during computations. They proposed mitigation strategies, such as memory access masking, to reduce the risk of such side-channel leaks, highlighting the ongoing need for effective countermeasures to secure HE systems against SCAs

2. Preliminaries

In this section, we provide foundational concepts essential for understanding the subsequent discussions on homomorphic encryption, side-channel attacks, and the vulnerabilities arising from their intersection.

2.1 Homomorphic Encryption

Homomorphic encryption (HE) is a cryptographic technique that permits computations to be performed directly on encrypted data without requiring decryption. The result of such computations, when decrypted, yields the same outcome as if the operations were performed on the plaintext. This property is invaluable for preserving data privacy during processing.

2.1.1 Mathematical Foundation

A homomorphic encryption scheme is defined by a tuple of algorithms:

2. **Key Generation (KeyGen):** Produces a public and private key pair.
3. **Encryption (Enc):** Converts plaintext data into ciphertext using the public key.
4. **Decryption (Dec):** Recovers the plaintext from ciphertext using the private key.
5. **Evaluation (Eval):** Performs specific operations on ciphertexts, resulting in a new ciphertext that corresponds to the operation applied to the underlying plaintexts.

Formally, for a given operation ◦ \circ◦ and plaintexts m1m_1m1 and m2m_2m2 :

$$Dec(Eval(Enc(m1),Enc(m2)))=m1 \circ m2 \text{Dec}(\text{Eval}(\text{Enc}(m_1), \text{Enc}(m_2))) = m_1 \circ m_2 Dec(Eval(Enc(m1),Enc(m2)))=m1 ◦ m2$$

This equation ensures that operations on encrypted data are consistent with those on plaintexts.

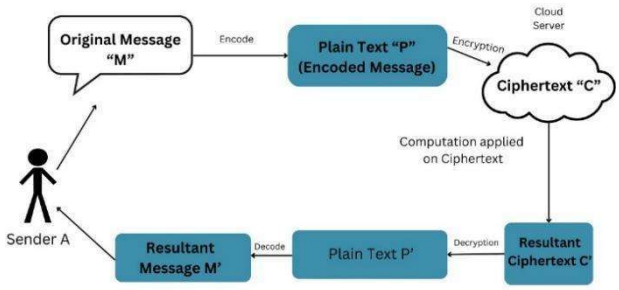

Figure 1: Homomorphic Encryption Architecture Diagram

2.1.2 Types of Homomorphic Encryption

- **Partially Homomorphic Encryption (PHE):** Supports a single type of operation (either addition or multiplication) on ciphertexts.
- **Somewhat Homomorphic Encryption (SHE):** Allows a limited number of both addition and multiplication operations.
- **Fully Homomorphic Encryption (FHE):** Enables arbitrary computations, including any number of additions and multiplications, on ciphertexts.

2.2 Side-Channel Attacks

Side-channel attacks (SCAs) exploit unintended information leakage from the

physical implementation of cryptographic systems. Instead of targeting theoretical weaknesses in algorithms, SCAs leverage observable phenomena such as timing information, power consumption, electromagnetic emissions, or acoustic signals to infer sensitive data.

2.2.1 Common Types of Side-Channel Attacks

- **Timing Attacks:** Analyze the time taken to execute cryptographic algorithms to deduce information about the secret keys.
- **Power Analysis Attacks:** Monitor power consumption patterns during cryptographic operations to extract secret information.
- **Electromagnetic Attacks:** Measure electromagnetic emissions from devices during computation to gather confidential data.
- **Acoustic Cryptanalysis:** Utilize sounds produced by hardware components during processing to retrieve sensitive information.

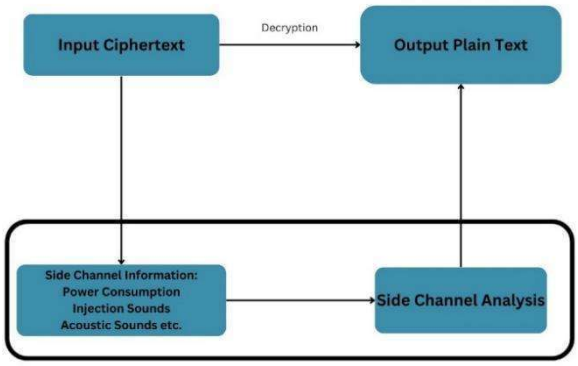

Figure 2: Side Channel Attack

2.3 Homomorphic Encryption Vulnerabilities to Side-Channel Attacks

While homomorphic encryption schemes are theoretically designed to be secure, their practical implementations can be susceptible to side-channel attacks. The complexity of HE operations often results in prolonged computation times and increased power consumption, providing a larger surface for SCAs. For instance, during the evaluation phase, variations in power usage or execution time can inadvertently leak information about the processed ciphertexts, potentially allowing attackers to reconstruct the underlying plaintexts or even recover secret keys.

Understanding these foundational concepts is crucial for analyzing the security of homomorphic encryption systems and developing effective countermeasures against side-channel attacks.

3. Side Channel Vulnerability Threats in Homomorphic Encryption

3.1 Types of Side-Channel Attacks in Homomorphic Encryption

SCAs leverage observable characteristics such as timing information, power consumption, or electromagnetic emissions to infer sensitive information. In the context of HE, these attacks can be particularly concerning:

- **Timing Attacks**: Variations in execution time during HE operations can reveal information about the processed data.
- **Power Analysis Attacks**: Monitoring power consumption patterns during HE computations can lead to the extraction of secret keys or plaintexts.
- **Electromagnetic Attacks**: Electromagnetic emissions during HE operations can be analyzed to uncover confidential information.

3.2 Documented Vulnerabilities

Recent studies have highlighted specific vulnerabilities in HE implementations:

- **Microsoft SEAL Library**: Research demonstrated that side-channel leakages in the Microsoft SEAL HE

- library could be exploited to steal encryption keys during the key generation phase. The attack abuses leakage in the number theoretic transform (NTT) algorithm, which is integral to the library's operations.

- **Single-Trace Attacks**: Another study revealed that single-trace side-channel attacks could extract entire plaintext messages from the SEAL library's encryption phase with just one power measurement, showcasing the feasibility of such attacks in real-world scenarios.

3.3 Threats in HE aiding to side-channel attacks

3.3.1. Side-Channel Attacks (SCAs)

- Side-channel attacks exploit unintended leakage from physical implementations of cryptographic algorithms, such as timing variations, power consumption, or electromagnetic emissions during computations.
- **Example**: Single-trace power analysis attacks can extract plaintexts or encryption keys from HE schemes, such as Microsoft SEAL, by analyzing power consumption during encryption or evaluation phases.
- **Impact**: SCAs bypass theoretical security guarantees, leaking sensitive information without directly attacking the cryptographic algorithms.

3.3.2. Implementation Bugs

- Errors in implementing HE schemes, such as flawed arithmetic or insecure library usage, can introduce exploitable vulnerabilities.
- **Example**: Incorrect parameter settings in fully homomorphic encryption (HE) schemes may lead to ciphertext malleability or reduced security levels.
- **Impact**: Implementation bugs can compromise cryptographic guarantees and expose data to unauthorized access.

3.3.3 Parameter Misconfiguration

HE schemes rely on secure parameter settings, such as key size, noise levels, and ciphertext modulus. Incorrect configurations can undermine security.

- **Example**: Choosing a low noise budget in HE schemes may result in decryption errors or easier attacks by adversaries.
- **Impact**: Inadequate parameter selection compromises both functionality and security.

3.3.4. Resource Exhaustion Attacks

- HE operations are computationally intensive, and attackers can exploit this to execute denial-of-service (DoS) attacks.
- **Example**: Sending repeated requests for complex HE evaluations can overload system resources.
- **Impact**: Such attacks disrupt system availability, particularly in cloud-based HE implementations.

3.3.5. Data Leakage During Operations

Intermediate data generated during HE operations may unintentionally reveal information about the encrypted data.

- **Example**: Bootstrapping (a noise-reduction operation in HE) can potentially leak sensitive information if not securely implemented.

- **Impact**: Leaked intermediate values undermine data confidentiality.

- **3.3.6. Chosen-Ciphertext Attacks (CCA)**

In chosen-ciphertext attacks, adversaries manipulate ciphertexts to infer information about the plaintext or encryption keys.

- **Example**: Modifying ciphertexts and observing the decryption effects can help attackers reconstruct sensitive data.
- **Impact**: Improper validation or padding mechanisms in HE implementations can make them susceptible to CCA.

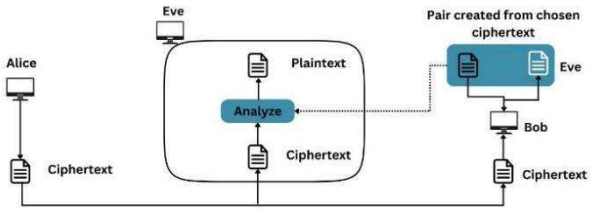

Figure 3: Chosen Ciphertext Attack

3.3.7. Cross-Platform Vulnerabilities

- Variations in hardware or software platforms introduce unique vulnerabilities in HE systems.
- **Example**: An HE library optimized for one hardware may leak side-channel information when deployed on less secure devices.
- **Impact**: Platform-specific weaknesses provide attackers with unexpected attack vectors.

3.3.8. Key Management Challenges

Secure generation, storage, and distribution of HE keys are critical. Improper key management practices can result in unauthorized access.

- **Example**: Exposing private keys in shared environments can compromise encrypted data for multiple users.
- **Impact**: Poor key management directly undermines HE system security.

3.3.9. Quantum Threats

Future advancements in quantum computing pose a significant threat to HE schemes.

- **Example**: Shor's algorithm could break cryptographic primitives based on integer factorization or discrete logarithms.
- **Impact**: The confidentiality of encrypted data may not be guaranteed in a post-quantum era.

4. Countermeasures To Combat Side Channel Threats In Homomorphic Encryption

To address the vulnerabilities associated with homomorphic encryption (HE), several countermeasures have been proposed and implemented. These measures focus on enhancing the security and resilience of HE systems against known threats, ensuring both theoretical robustness and practical applicability.

Below are the key countermeasures:

4.1. Mitigating Side-Channel Attacks (SCAs)

- **Constant-Time Implementations**: Design algorithms to execute in constant time, independent of input values, to prevent timing-based leakages.
- **Power and Electromagnetic Analysis Resistance**: Employ masking techniques or hardware shields to obscure power consumption and electromagnetic emissions.
- **Secure Hardware Environments**: Deploy HE computations on tamper-resistant hardware or trusted execution environments (TEEs) to reduce susceptibility to SCAs.

4.2. Preventing Implementation Bugs

- **Rigorous Testing**: Conduct extensive testing of HE implementations using both automated and manual methods to identify and rectify potential flaws.
- **Formal Verification**: Apply formal methods to verify the correctness of HE algorithms and their implementation.
- **Library Security Audits**: Perform regular audits of HE libraries to ensure compliance with secure coding practices.

4.3. Ensuring Proper Parameter Configuration

- **Guidelines for Parameter Selection**: Follow established standards, such as those outlined by NIST or HomomorphicEncryption.org, to choose secure and optimal parameters.
- **Dynamic Parameter Adjustment**: Implement adaptive systems to monitor and adjust parameters based on the computational context and noise levels.

4.4. Preventing Resource Exhaustion Attacks

- **Rate Limiting**: Limit the number of HE operations a system can perform within a specific timeframe to prevent overload.
- **Efficient Algorithms**: Optimize HE algorithms to reduce computational overhead and mitigate susceptibility to denial-of-service (DoS) attacks.
- **Cloud-Based Load Balancing**: Use load-balancing techniques in cloud-based HE systems to distribute computational tasks and avoid resource bottlenecks.

4.5. Preventing Data Leakage During Operations

- **Secure Bootstrapping**: Employ advanced bootstrapping techniques that minimize leakage, such as noise-hiding mechanisms or encryption masking.
- **Randomized Processing**: Introduce randomness into intermediate computations to obscure patterns that could reveal information.

4.6. Mitigating Chosen-Ciphertext Attacks (CCA)

- **Robust Padding Schemes**: Use strong padding schemes, such as Optimal Asymmetric Encryption Padding (OAEP), to prevent ciphertext manipulation.
- **Validation of Inputs**: Ensure strict validation of ciphertexts before processing to detect and reject **malformed inputs.**
- **Authenticated Encryption**: Combine HE with authenticated encryption to detect and prevent unauthorized ciphertext alterations.

4.7. Addressing Cross-Platform Vulnerabilities

- **Platform-Specific Optimization**: Tailor HE implementations for specific hardware platforms to ensure security across varying architectures.
- **Secure Deployment Guidelines**: Provide clear instructions for deploying HE libraries securely on diverse platforms.

4.8. Strengthening Key Management

- **Secure Key Storage**: Use hardware security modules (HSMs) or secure enclaves to store keys safely.
- **Robust Key Exchange Protocols**: Implement secure key exchange mechanisms, such as Diffie-Hellman or post-quantum cryptographic methods, to protect key transmission.

- **Periodic Key Rotation**: Regularly update and rotate encryption keys to reduce the risk of compromise.

4.9. Preparing for Quantum Threats

- **Post-Quantum Cryptography Integration**: Transition to HE schemes that rely on quantum-resistant assumptions, such as lattice-based cryptography.
- **Hybrid Cryptographic Solutions**: Combine HE with classical and post-quantum algorithms to ensure long-term data confidentiality.
- **Continuous Monitoring of Quantum Advancements**: Monitor developments in quantum computing to proactively update HE systems against emerging threats.

6. Conclusion

Homomorphic encryption (HE) presents a powerful tool for secure data processing, particularly in cloud computing and privacy-preserving applications. However, as the literature demonstrates, the security of HE systems is not immune to side-channel attacks (SCAs). Research has uncovered various vulnerabilities in widely used HE libraries, such as Microsoft SEAL, revealing that attackers can exploit side-channel leakages to extract sensitive information, including plaintext data and secret keys. These vulnerabilities are significant as they can undermine the privacy guarantees provided by HE, especially in real-world scenarios where attackers may have access to physical devices or computational traces.

The studies reviewed in this paper highlight the importance of addressing side-channel risks in HE systems. While existing work has identified several forms of side-channel leakages, including those related to power consumption, timing variations, and memory access patterns, there remains a need for more robust countermeasures. Techniques such as constant-time implementations, noise injection, and memory access masking show promise in mitigating these threats, but there is still much work to be done to achieve a comprehensive and efficient defence strategy.

As HE continues to evolve, researchers must focus on developing secure and efficient implementations that can withstand side-channel attacks without sacrificing performance. Future research should explore novel techniques to enhance the resilience of HE against SCAs, ensuring its practical adoption in sensitive and privacy-critical applications. Moreover, standardization of security measures across different HE implementations will play a key role in ensuring the consistent protection of encrypted data in the face of emerging attack vectors.

In conclusion, while homomorphic encryption remains a promising cryptographic tool, its security must be strengthened against side-channel threats to guarantee its effectiveness in safeguarding data privacy.

References

Books and Journal Articles:

1. **"Cryptographic Engineering" by Çetin Kaya Koç**: This book provides an in-depth exploration of cryptographic algorithms and their implementations, including discussions on side-channel attacks and defenses.

2. **"Leaking Secrets in Homomorphic Encryption with Side-Channel Attacks" by Furkan Aydin and Aydin Aysu**: Published in the *Journal of Cryptographic Engineering*, this article demonstrates side-channel leakages in the Microsoft SEAL homomorphic encryption library, highlighting

vulnerabilities during the key generation phase.

3. **"Exposing Side-Channel Leakage of SEAL Homomorphic Encryption Library" by Furkan Aydin et al.**: This paper reveals new side-channel leakages in the Microsoft SEAL library, focusing on vulnerabilities that could be exploited to extract sensitive information.

YouTube Videos:

4. **"Single-Trace Side-Channel Attack on SEAL Homomorphic Encryption Library"**: In this talk, Furkan Aydin discusses the first side-channel attack on homomorphic encryption, demonstrating how power analysis can be used to extract plaintext from encrypted data.
5. **"Hardware Security Tutorial - Part 4 - Side Channel Attacks"**: Presented by Prof. Todd Austin from the University of Michigan, this tutorial provides an overview of side-channel attacks, including those relevant to cryptographic implementations

Blogs and Online Articles:

6. **"Homomorphic Encryption Targeted by a New Side-Channel Attack Campaign"**: This article discusses recent research demonstrating that even advanced encryption methods like homomorphic encryption are susceptible to side-channel attacks, emphasizing the need for robust security measures.

7. **"Researchers Demonstrate They Can Steal Data During Homomorphic Encryption"**: This blog post highlights vulnerabilities in homomorphic encryption that could allow attackers to extract data during the encryption process, underscoring the importance of implementing effective countermeasures.

Government and Private Reports:

8. **"Lifting the Veil on Homomorphic Encryption" by the Office of the Privacy Commissioner of Canada**: This report provides an overview of homomorphic encryption, its benefits, and potential vulnerabilities, including discussions on.

9. **"Toward Scalable Fully Homomorphic Encryption Through Light Trusted Computing Assistance"**: This research paper explores enhancing the scalability of fully homomorphic encryption by integrating trusted computing technologies, addressing performance and security challenges.

10. Aydin, Furkan & Aysu, Aydin. (2022). Exposing Side-Channel Leakage of SEAL Homomorphic Encryption Library. 10.1145/3560834.3563833.

11. Pal, A., Swaminathan, K., Pal, S., & San Miguel, J. Characterizing Memory Side Channels in FHE Applications.

12. F. Aydin, E. Karabulut, S. Potluri, E. Alkim and A. Aysu, "RevEAL: Single-Trace Side-Channel Leakage of the SEAL Homomorphic Encryption Library," 2022 Design, Automation & Test in Europe Conference & Exhibition (DATE), Antwerp, Belgium, 2022, pp. 1527-1532, doi: 10.23919/DATE54114.2022.9774724.

13. Marcolla, Chiara & Sucasas, Victor & Manzano, Marc & Bassoli, Riccardo &

Fitzek, Frank & Aaraj, Najwa. (2022). Survey on Fully Homomorphic Encryption, Theory, and Applications. Proceedings of the IEEE. PP. 1-38. 10.1109/JPROC.2022.3205665.

Academic and Conference Papers

Zhao, F., Wang, X. S., et al. (2020)
"Cryptanalysis of Machine Learning with Homomorphic Encryption and Applications to Medical Data Privacy"
In *USENIX Security Symposium*
https://www.usenix.org/conference/usenixsecurity20

Zhang, Z., Wang, X., & Wang, C. (2021)
"Side-Channel Vulnerabilities in Fully Homomorphic Encryption Libraries"
In *Proceedings of the 2021 ACM Workshop on Privacy in the Electronic Society (WPES)*
DOI: 10.1145/3471984.3473042

Chen, Y., et al. (2017)
"Side-channel leakage in matrix multiplication and its implication for lattice-based cryptography"
In *ACM CCS*
DOI: 10.1145/3133956.3134092

Liu, S., Wang, H., et al. (2016)
"A survey on side-channel attacks and countermeasures"
Journal of Cryptographic Engineering, 8(4), 315–339
DOI: 10.1007/s13389-016-0122-0

Yonetani, R., et al. (2021)
"Cache Side-channel Attacks and Countermeasures in FHE"
In *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*
IEEE Xplore

Gentry, C. (2009)
"Fully Homomorphic Encryption Using Ideal Lattices"
Ph.D. Thesis, Stanford University
https://crypto.stanford.edu/craig

Smart, N., & Vercauteren, F. (2014)
"Fully Homomorphic SIMD Operations"
In *Designs, Codes and Cryptography*
Also available in the Springer book *Homomorphic Encryption and Applications*, 2014.

Bhasin, S., & Regazzoni, F. (Eds.) (2019)
"Hardware Side-Channel Attacks: Experimental Analysis of Real Devices"
Springer International Publishing.
ISBN: 978-3-030-30618-7

Alkim, E., Bindel, N., et al. (2020)
"A New Hope: Efficient Post-Quantum Key Exchange"
USENIX Security
Discusses masking and side-channel resistance for lattice-based crypto.

Bayrak, A. G., Regazzoni, F., et al. (2013)
"Spook: Side-channel protection through obscurity"
In *ACM/IEEE DAC*
DOI: 10.1145/2463209.2488764

Chari, S., Rao, J. R., & Rohatgi, P. (2002)
"Template Attacks"
In *CHES 2002: Cryptographic Hardware and Embedded Systems*
Lecture Notes in Computer Science, Springer.

Schindler, W. (2009)
"A Timing Attack against RSA with the Chinese Remainder Theorem"
In *Cryptographic Hardware and Embedded Systems – CHES 2000*

Suggestions for Practical Exploration

Review the Microsoft SEAL library and its security notes

https://github.com/microsoft/SEAL

- Explore **PALISADE** and **HElib** to test side-channel protections
https://palisade-crypto.org
https://github.com/homenc/HElib

Check out **OpenFHE**, the open-source successor to PALISADE, with improved FHE APIs
https://openfhe.org